Міністерство освіти і науки України

Харківський національний університет ім. В. Н. Каразіна

Факультет комп’ютерних наук

**КУРСОВА РОБОТА**

з дисципліни «Комп’ютерна схемотехніка та архітектура комп’ютерів»

Тема «Периферія Timer. Робота через переривання і DMA»

|  |  |
| --- | --- |
| Оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Члени комісії:  \_\_\_\_\_\_\_\_\_\_\_ Рало О.М.  \_\_\_\_\_\_\_\_\_\_\_Осипчук А.В.  \_\_\_\_\_\_\_\_\_\_\_ Стервоєдов М. Г. | Виконав:  студент 3 курсу, групи КС-32 |
| Спеціальності:  122 «Комп’ютерні науки»  Безрук Юрій Русланович |
|  |

Харків – 2020

**ЗМІСТ**

[ВСТУП 3](#_Toc59482796)

[РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ 4](#_Toc59482797)

[РОЗДІЛ 2 ПЕРИФЕРІЯ TIMER 5](#_Toc59482798)

[2.1 Загальні характеристики 5](#_Toc59482799)

[2.2 Види таймерів 6](#_Toc59482800)

[РОЗДІЛ 3 ПРАКТИЧНЕ ЗАСТОСУВАННЯ 9](#_Toc59482801)

[3.1 Застосування переривань 9](#_Toc59482802)

[3.2 Застосування DMA 10](#_Toc59482803)

[РОЗДІЛ 4 ПРОГРАМНА РЕАЛІЗАЦІЯ 12](#_Toc59482804)

[4.1 Реалізація таймера з перериваннями 12](#_Toc59482805)

[4.2 Реалізація таймера з DMA 15](#_Toc59482806)

[ВИСНОВКИ 18](#_Toc59482807)

[ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 19](#_Toc59482808)

# ВСТУП

Таймер – один з найважливіших для мікроконтролера пристроїв, який застосовується для вирішення безлічі завдань: відлік часу, планування дій, генерація сигналів, вимірювання параметрів сигналів та ін. Розробники пристроїв на базі мікроконтролерів активно використовують таймери, а розробники мікроконтролерів, в свою чергу, намагаються найбільш повно задовольнити потреби користувачів, надаючи їм в розпорядження різні за складністю та можливостям таймери.

Так, типовий представник мікроконтролерів STM32 містить близько десяти таймерів. В їх число входять як прості таймери з базовим набором функцій, так і таймери вельми складні і багатофункціональні.

У STM32 мікроконтролерах дійсно багато таймерів, вони дійсно мають дуже велику кількість функцій. Ці таймери відрізняються один від одного складом і своїми можливостями, що ще більш ускладнює їх вивчення. Опису таймерів присвячені сотні сторінок тексту в довідковому керівництві. Значно спрощує життя той факт, що всі таймери в STM мають єдиний принцип побудови і управління.

# РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ

В ході виконання даної роботи буде розглянуто основні характеристики периферії таймер, види таймерів, їх застосування за допомогою переривань та DMA та деякі підходи програмної реалізації взаємодії с даною периферією.

Отже, об’єктом дослідження є периферія Timer.

Предметом дослідження є методи та засоби використання периферії на практиці та роботи з перериваннями та DMA на мікроконтролері stm32f4.

Метою роботи є покращення навичок роботи з мікроконтролером сімейства STM32 в області роботи з даною периферією, периферією NVIC та DMA.

# РОЗДІЛ 2 ПЕРИФЕРІЯ TIMER

## 2.1 Загальні характеристики

Таймер - пристрій, що відмірює заданий інтервал часу, після закінчення якого він виробляє певний сигнал.

Коли таймер є частиною мікроконтролера, способом сигналізації може бути, наприклад, переривання. Зазвичай таймер працює в періодичному режимі, тобто не зупиняється після формування сигналу, а починає відразу ж відміряти наступний інтервал такої ж тривалості. В результаті таймер генерує сигнали (переривання) з певною частотою, через задані інтервали часу.

Сигнальний режим має на увазі те, що таймер відміряє заданий інтервал часу, по його закінченні генерує сигнал і зупиняється. Таймер може мати апаратну підтримку сигнального режиму, або цей режим можна легко реалізувати програмно (досить відключити таймер в обробнику переривання або виконати необхідні дії всередині обробника тільки при першому виклику).

Технічно таймер будується на основі лічильника, який підрахує імпульси деякого тактового сигналу. Зазвичай сигнал про спрацювання таймера формується при переповненні лічильника. Як правило, є можливість управляти модулем перерахунку лічильника таймера (задавати кількість імпульсів між двома послідовними переповнення лічильника). Це дозволяє при фіксованій частоті тактового сигналу на вході лічильника встановлювати необхідну частоту генерації сигналів від таймера. Крім того, зазвичай є програмний доступ до лічильника таймера, за поточним значенням якого можна з високою роздільною здатністю визначати час, що минув з моменту останнього переповнення лічильника.

Таймер дає можливість мікроконтролеру і програмі вести відлік часу, що визначає два основних напрямки застосування: для виконання необхідних дій в задані моменти часу і для визначення (фіксації) моментів, коли відбулися деякі цікаві для нас події.

До першого напряму застосування таймера можна віднести, наприклад, управління зовнішніми пристроями за заздалегідь запланованим розкладом, відповідно до якого в необхідні моменти часу програма посилає сигнали до контрольованих пристроїв. Інтервал часу між двома сусідніми запланованими моментами часу може варіюватися в широких межах - від одиниць періодів тактового сигналу до практично нескінченно (годинник, добу, роки). Незалежно від тривалості інтервалу, його можна поставити з високою точністю і з дуже високою роздільною здатністю в часі – до періоду тактового сигналу.

Здатність таймера відміряти дуже малі інтервали часу відкриває можливість застосування таймера для генерації сигналів: цифрових (для цього достатньо виконувати перемикання стану зовнішнього виведення відповідно до закону зміни синтезованого сигналу) або аналогових (за рахунок спільної роботи таймера і DAC).

Також таймер можна використовувати для організації середовища виконання програми. Мова йде, перш за все, про реалізацію багатозадачності шляхом перемикання між виконуваними завданнями по сигналу таймера.

Не менш важливим є інший напрямок використання таймера – для фіксації моменту настання подій. Це може бути: ведення журналу подій, вимірювання параметрів зовнішніх сигналів (тривалість імпульсу, період сигналу) і т.д. Сюди можна віднести, знову ж таки, рішення задач управління, тепер уже в тих випадках, коли реакція на зовнішній сигнал залежить від моменту його надходження.

## 2.2 Види таймерів

У мікроконтролерах Cortex-M, до яких відносяться і мікроконтролери STM32, завжди є, принаймні, один таймер - системний таймер, наявність якого гарантується вимогами архітектури. Стандартизація системного таймера на рівні архітектури істотно спрощує його використання. Як мінімум, це забезпечує переносимість коду, що використовує цей таймер між різними мікроконтролерами. Переносимість особливо важлива для коду операційних систем, так що не дивно, то, що системний таймер зазвичай задіяний для потреб операційної системи, якщо вона використовується, звичайно.

Крім системного таймера, мікроконтролери STM32 мають сторожовий таймер (і навіть не один). Сторожові таймери мають досить вузьку спеціалізацію - вони використовуються для автоматичної перезавантаження в разі зависання програми.

Крім того, мікроконтролери STM32 мають безліч універсальних таймерів, які можна використовувати на свій розсуд. На відміну від системного, вони є не частиною ядра, а периферійними пристроями мікроконтролера. Периферійні пристрої таймерів можна класифікувати наступним чином:

• Просунуті таймери, такі як TIM1 і TIM8.

• GP-таймери, на зразок TIM2 і TIM3.

• Таймери з полегшеною конфігурацією, такі як TIM9, TIM10, TIM12 і TIM16.

• Базові таймери, такі як TIM6 і TIM7.

Базові таймери є найпростішими, відповідно до своєї назви мають лише набір базових функцій. Таймери з розширеним управлінням – найскладніші і мають найбільшу кількість реалізованих функцій.

Для таймерів в мікроконтролерах сімейства STM32 використовується наступна система іменування: назва починається з префікса TIM, за яким слід номер таймера: TIM1, TIM2, і т.д. Важливим є те, що номер таймера визначає його тип. І в різних мікроконтролерах таймери з однаковими номерами зазвичай сумісні (однаково влаштовані, мають однаковий набір функцій і управляються однаковим чином за допомогою однакових наборів регістрів; що, однак, не звільняє від необхідності уважного ознайомлення з довідковим керівництвом: Не гарантується повна відсутність відмінностей). Такий підхід дуже зручний, оскільки дозволяє легко переходити від одного мікроконтролера до іншого - як в сенсі вивчення, так і при перенесенні коду. Використовувана в мікроконтролерах STM система іменування таймерів має на увазі те, що нумерація не обов'язково повинна бути послідовною, вона має пропуски, відповідні відсутнім в конкретної моделі мікроконтролера таймерам. Тому, наприклад, якщо мікроконтролер має таймер з максимальним номером 17 (TIM17), це ще не означає, що всього є 17 універсальних таймерів TIM1..TIM17 – таймери з якимись номерами будуть відсутні.

# РОЗДІЛ 3 ПРАКТИЧНЕ ЗАСТОСУВАННЯ

## 3.1 Застосування переривань

Переривання – це механізм, який дозволяє апаратного забезпечення повідомляти про настання важливих подій у своїй роботі. У момент, коли відбувається переривання, процесор перемикається з виконання основної програми на виконання відповідного обробника переривань. Як тільки виконання обробника завершено, триває виконання основної програми з місця, в якому вона була перервана.

Для використання переривань необхідно спочатку налаштувати регістр, який називається Nested Vector Interrupt Controller (NVIC), вбудований контролер вектора переривань. Даний регістр є стандартною частиною архітектури ARM і зустрічається на всіх процесорах, незалежно від виробника. NVIC розроблений таким чином, що затримка переривання мінімальна. NVIC підтримує вбудовані переривання з 16-ю рівнями пріоритету.

В процесі написання програми ми можемо сконфігурувати таймер таким чином, щоб сигнал, який він надсилає по закінченню свого періоду надходив модулю NVIC і той генерував переривання, яке ми зможемо обробити у своїй бізнес-логіці.

Для підключення другого таймеру до периферії NVIC потрібно задати значення TIM2\_IRQn полю NVIC\_IRQChannel. TIM2 підключений до шини APB1, тому для його ініціалізації потрібно спочатку активувати його на периферії матриці шин RCC.

Для задання періоду таймера використовують поля TIM\_Prescaler (попередній дільник) та TIM\_Period (множник періоду). Розрахунок виконується наступним чином: в попередній дільник встановлюють значення для зменшення частоти таймеру. Початкове значення частоти – це значення на якому працює шина, до якої підключено таймер. Це значення розділиться на число, яке встановлено в TIM\_Prescaler і результат буде частотою таймера. Оскільки частота – обернена до періоду величина, можемо дізнатися період спрацювання таймеру. Для корекції періоду використовують додатковий множник TIM\_Period, який збільшує період у величину разів, що в нього встановлено.

Оскільки таймер є лічильником, важливе значення має режим спрацьовування сигналу, так режим «лічіння». Типовим прикладом є використання лічіння догори (TIM\_CounterMode\_Up) з прапором на переповнення (TIM\_IT\_Update). Це означає, що значення таймеру буде інкрементуватися поки не досягне періоду, після чого буде сгенеровано сигнал.

Цей сигнал буде надаватися модулю NVIC, який сгенерує переривання. Програміст має його обробити за допомогою обробника переривань. Під час обробки обов’язково слід очистити значення прапору TIM\_IT\_Update у разі якщо воно було сгенеровано по ньому для того, щоб переривання не виконувалося у невідповідний час.

## 3.2 Застосування DMA

DMA (Direct Memory Access) - технологія прямого доступу до пам'яті. Ця технологія дозволяє швидко і без використання центрального процесора пересилати дані з однієї області пам'яті в іншу. При цьому для такої пересилки замість ЦП використовується свій спеціальний контролер, який називається контролером DMA. Тобто вся ця робота буде відбуватися без будь-якої участі ЦПУ, все буде робити модуль DMA, а центральний процесор у цей час має змогу виконувати свою бізнес-логіку.

Незважаючи на те, що ЦП і DMA працюють незалежно один від одного, DMA може призупиняти доступ ЦП до системної шини на кілька тактів у випадках коли вони обидва намагаються звернутися до одним і тим же адресами пам'яті.

DMA може передавати дані і у регістри таймера в тому числі. Це можна використовувати, наприклад, для генерації широко-імпульсної модуляції сигналів (ШІМ), яка використовується для плавного розпалу елементів, таких як діоди чи навіть двигуни. В цьому випадку таймер буде генерувати ШІМ в потрібний йому канал, а дані для каналу зможе передавати йому DMA, в обхід центрального процесору.

В DMA для цього необхідно вибрати режим роботи пам’ять у периферію, і надати необхідні адреси даних та самої периферії. Додатково слід обрати розмір буфера та режим роботи контролеру прямого доступу до пам’яті.

# РОЗДІЛ 4 ПРОГРАМНА РЕАЛІЗАЦІЯ

## 4.1 Реалізація таймера з перериваннями

Для прикладу роботи з таймером реалізуємо такий функціонал програмно: контролер буде циклічно запалювати діоди на платі по колу з інтервалом в одну секунду. Тобто спочатку запалиться один з діодів, і буде світити секунду, після чого згасне і запалиться наступний по колу. Коли згасне останній, усі діоди на секунду запаляться одночасно, після чого процес буде повторюватися. Перейдемо до реалізації, використовуючи таймер загального призначення TIM2.

Для початку реалізуємо логіку програми – для цього створюємо перечислення State, яке буде вирішувати режим – який діод зараз палатиме. Відповідно, її поля будуть LED4, LED3, LED5, LED6, ALL. В функції logic() реалізуємо перемикач станів – в залежності від поточного значення змінної перечислення state, гасимо попередній та запалюємо наступний світлодіод. У випадку ALL запалюємо всі діоди разом, а у виключній ситуації встановлюємо початкове значення.

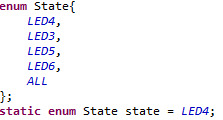


Рисунок 1 – реалізація перечислення стану

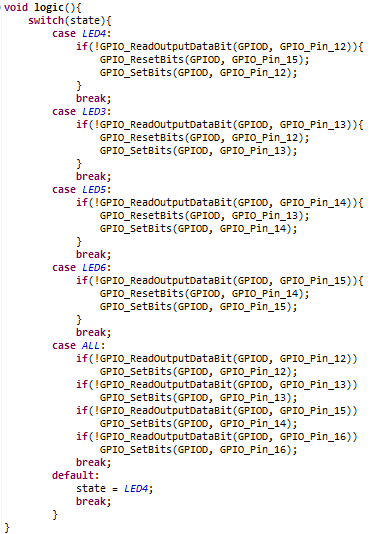


Рисунок 2 – логіка

Далі вказуємо основні налаштування для роботи з периферіями у функціях INTTIM\_Config() та GPIO\_Config().

В функції GPIO\_Config() конфігуруємо периферію ніжок контролера (пінів). Для цього активуємо шину AHB1, встановлюємо піни, на яких знаходяться діоди (12-15) на режим виходу.

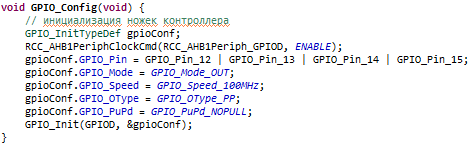


Рисунок 3 – ініціалізація периферії GPIO

В INTTIM\_Config() спочатку налаштовуємо периферію NVIC, вказуючи їй канал другого таймеру. Вказуємо первинний пріоритет переривання як 0 (найвищий), а вторинний як 1. Далі налаштовуємо сам таймер. Для цього необхідно розрахувати, які числа підібрати для попереднього дільника та множника. Рахуємо так: сам процесор працює на частоті 168МГц. На шині APB1, до якої підключаємо таймер, підключено дільник ½, що означає, що її частота – 84МГц, або 84 000 000 Гц. Для того, щоб отримати період в одну секунду, встановлюємо попередній дільник таймеру 8400. Це означає, що таймер працюватиме та частоті 10кГц, а отже його період – 0, 0001 с. Тепер, для отримання періоду в одну секунду достатньо встановити множник TIM\_Period = 10000. Також встановлюємо режим лічіння догори і флаг на переповнення.

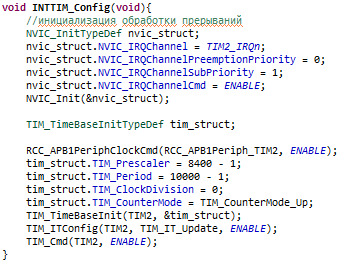


Рисунок 4 – конфігурація переривань та таймеру

Додаємо функцію-обробник переривань TIM2\_IRQHandler(). Вона буде автоматично викликатися при генерації нашого переривання. На вході одразу ж перевіряємо, що потрапили в обробник саме за тим прапором, який нам потрібен, очищаємо його та перемикаємо стан. У функції main() спочатку викликаємо функції ініціалізації, а далі у нескінченному циклі викликаємо функцію logic(). Таким чином, кожного разу коли генерується переривання, змінюється значення змінної state, що призводить до того, що в функції логіки програма заходить у інший кейс.

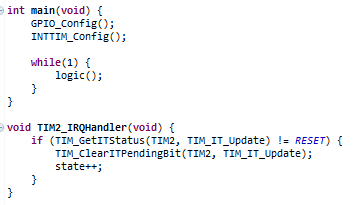


Рисунок 5 – функція main() та обробник переривань

## 4.2 Реалізація таймера з DMA

Нехай треба реалізувати таку програму: виведення сигналу шим на ніжку контролеру в фоновому для процесору режимі. Тобто без використання логіки, а з використанням DMA. Реалізуємо це таким чином.

Для початку слід налаштувати таймер на роботу з сигналом ШІМ. Для прикладу візьмемо четвертий таймер. Потрібні нам значення для генерації ШІМ встановлюємо відповідно у поля періоду и дільника. Налаштування таймеру робимо як і у минулий раз, за виключенням режиму, який тепер встановлюємо як TIM\_OCMode\_PWM1. Після цього налаштовуємо канали таймеру які хочемо використовувати.

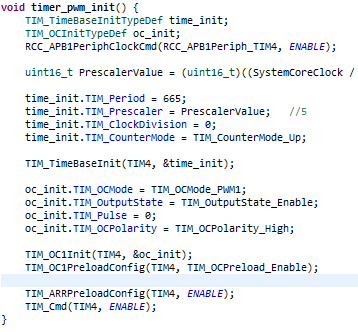


Рисунок 6 – налаштування ШІМ таймера

Далі готуємо далі для ШІМ-у: створюємо в пам’яті масив значень скважності, які будемо передавати в канал. після цього починаємо налаштування DMA. Активуємо шину AHB1, деініціалізуємо про всяк випадок потік DMA, з яким будемо працювати. Вказуємо конфігурації: початкова адреса периферії – адреса каналу таймера с ШІМ-ом., початкова адреса у пам’яті – адреса першого елементу масиву даних. Режим роботи – дані у периферію. Інкрементацію по периферії вимкнути, а по пам’яті – ввімкнути. Інші налаштування стандартні.

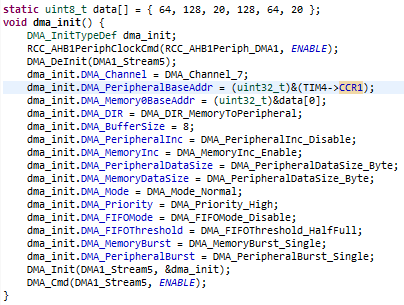


Рисунок 7 – налаштування DMA

Всі канали налаштовані. Після цього варто лише визвати функцію TIM\_DMA\_Cmd(TIM4, ENABLE), в будь-якому місці програми і модуль почне передачу даних з масиву в канал таймеру з ШІМ.

# ВИСНОВКИ

Отже, в даній курсовій роботі ми ознайомилися з периферією таймер, їх видами та призначенням, а також про роботу таймерів з перериваннями та DMA. Також були розроблені відповідні програми, що демонструють роботу таймерів різного призначення через різну периферію.

Можна зробити висновок про те, що периферія Timer є однією з найважливіших периферій мікроконтролера stm32, адже вона має досить багато різноманітних функцій і засобів використання, може працювати з модулями NVIC (переривання) та DMA (прямий доступ до пам’яті), а також генерувати сигнал ШІМ, використання якого дуже розповсюджене.

Використання переривань доцільне через свою асинхронність і можливість не відстежувати час у програмі, знаючи, що деяка функціональність буде виконана через вказаний проміжок часу.

Використання DMA розвантажує процесор і дозволяє досягти дуже високої швидкості передачі даних, якщо нам потрібно оперувати байтами даних при передачі з пам’яті в пам’ять, з пам’яті в периферію або з периферії в пам’ять без використання додаткової бізнес-логіки.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. STM32F4: INTERRUPT TIMER. [Електронний ресурс] – Режим доступу: http://amarkham.com/?p=29.
2. Reference manual. STM32F40xxx, STM32F41xxx, STM32F42xxx, STM32F43xxx advanced ARM-based 32-bit MCUs. [Електронний ресурс] – Режим доступу: http://www.st.com/web/en/resource/ technical/document/ reference\_manual/DM00031020.pdf.
3. Руководство по использованию обычных таймеров. [Електронний ресурс] – Режим доступу: http://microsin.net/programming/arm/an4776-general-purpose-timer-cookbook.html
4. Таймеры в микроконтроллерах STM. . [Електронний ресурс] – http://www. rotr.info/electronics/mcu/stm32f100/timers.htm